

Simulation Output Subsystem Status

B. Bush and K. Berkbigher

Los Alamos National Laboratory

Outline

- definition
- specification
- issues
- requirements
- design
- development
- usage
- status
- work underway

Definition

The output subsystem collects data from a running microsimulation, stores the data for future use, and manages the subsequent retrieval of the data.

Specification

- The simulation output subsystem will gather the data generated by simulations and provide access to it for other subsystems as soon as the data is received.
- The simulation output will be configurable and several predefined configurations will be provided.
- This subsystem will utilize the database subsystem to support metadata.
- It will support data distribution, data export, and archiving.
- Special provision will be made for dealing with the large amount of data generated by simulations.

Issues

- The CA microsimulation will generate large amounts of data on several computational nodes (CPNs) simultaneously.
- The computer communications network (ethernet, etc.) is required for simulation-related communication between CPNs; any other use of it will slow down the simulation.
- Users will want to specify what data will be collected, and when and where it will be collected.
- Users will want to retrieve only a portion of the complete data set, in order to perform analyses on data of interest.
- Users will want automated support for navigating through data sets.

Data Volume

- Approximately 30-60 bytes of data are needed to describe the state of a vehicle at any given time in the simulation.
- Data can conceivably be collected for each vehicle every time it is moved (currently, once per second).
- In the largest metropolitan areas, it is possible for 1,000,000 vehicles to be moving on the road network simultaneously.
- This means that a four-hour simulation could require 400-800 GB of storage, if all trajectory data is stored.
- *Conclusion:* The output subsystem has to efficiently store large amounts of data; it also must have the capability to *not* store data when it is not necessary to do so.

Requirements

- Data must be stored locally on each computational node, so that communication network traffic is minimized.
- Data must be retrieved globally, so that it can be analyzed anywhere.
- Several retrieval formats must be supported, so that post-processing is flexible.
- The simulation output subsystem must have a general interface, so that it can collect data from any TRANSIMS simulation, not just the current CA-based simulation.

Requirements (continued)

- Both as the data is collected and when it is retrieved, it must be possible to summarize the data:

- counting
- averaging
- accumulating
- binning
- statistical functions

This will reduce storage requirements and access time and minimize data transmission over the communication network.

- Metadata for each data set must be available, so that the data set is self-defining.
- The simulation output subsystem must have a “low overhead,” so that it does not unduly slow a running simulation.

TRANSIMS Software Architecture

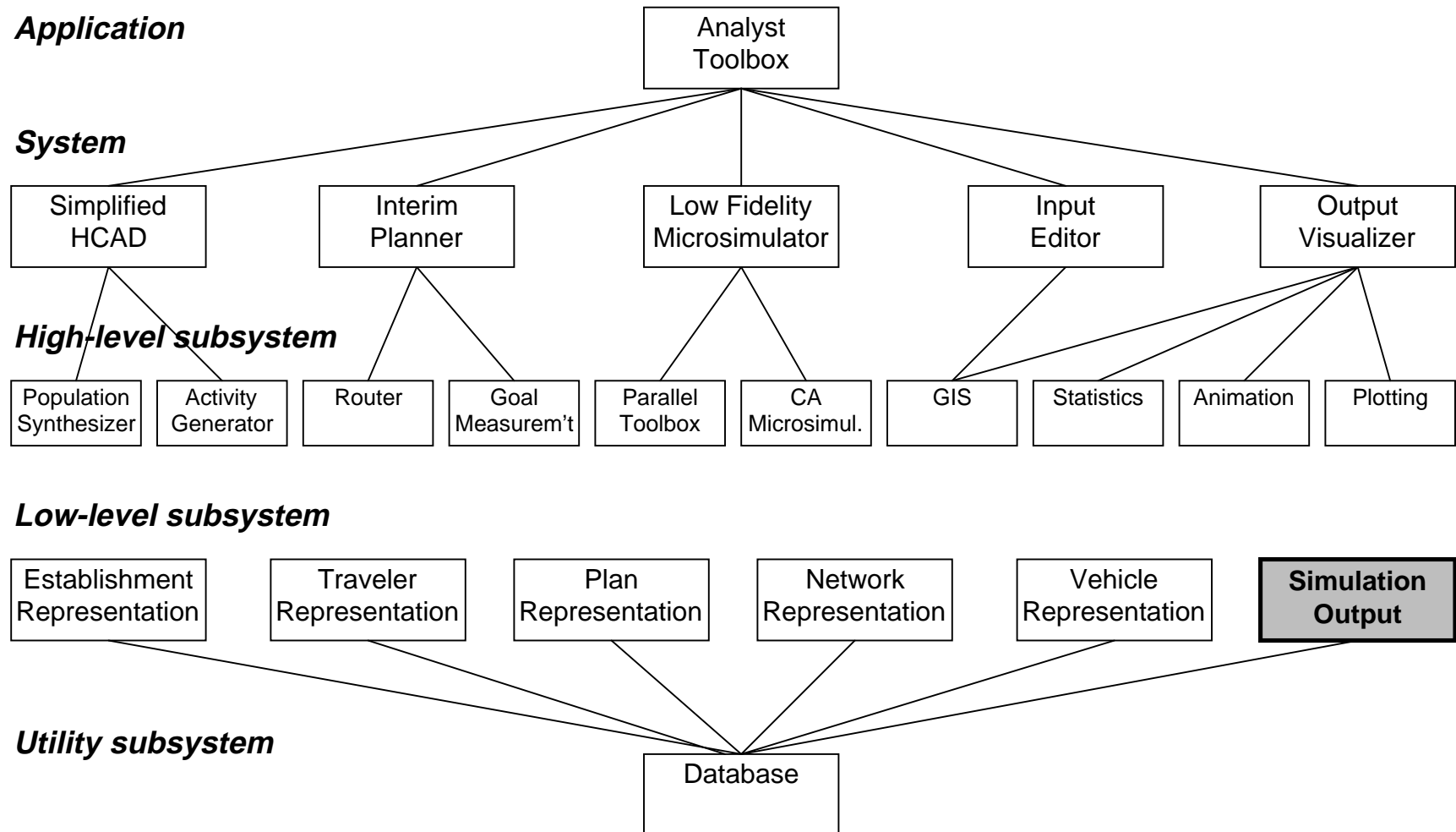
Application

System

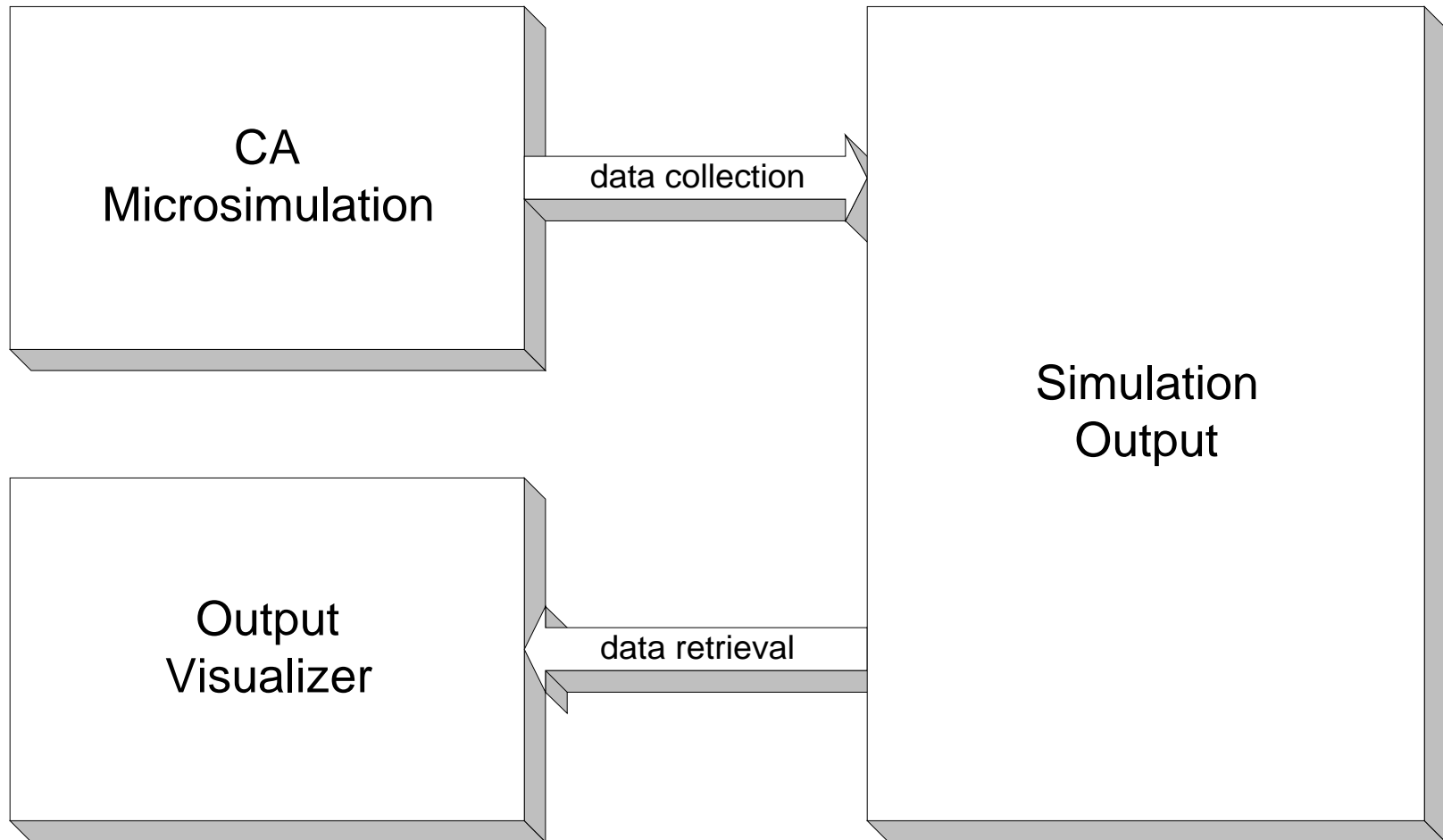
High-level subsystem

Low-level subsystem

Utility subsystem

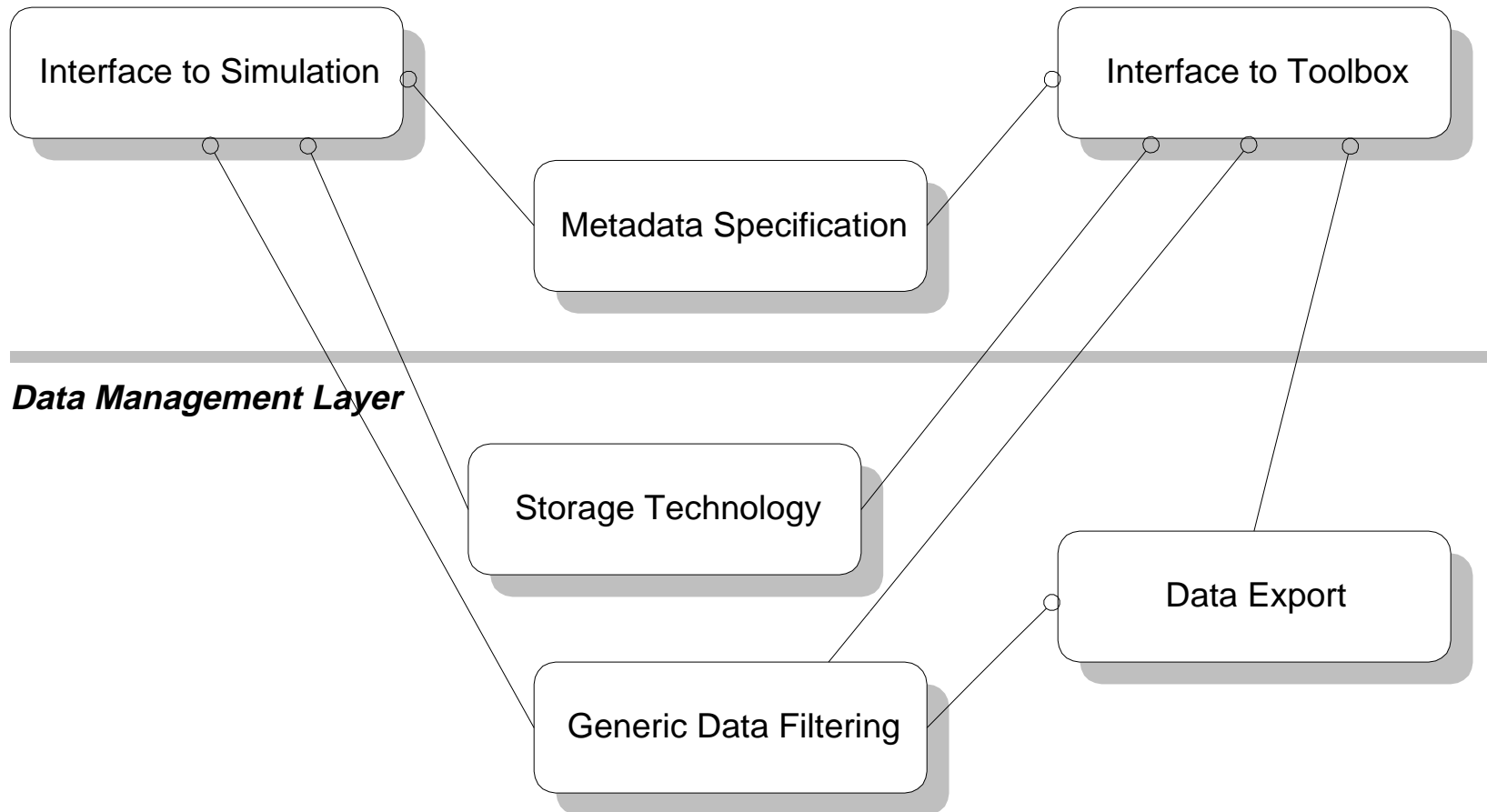


Connections between Subsystems

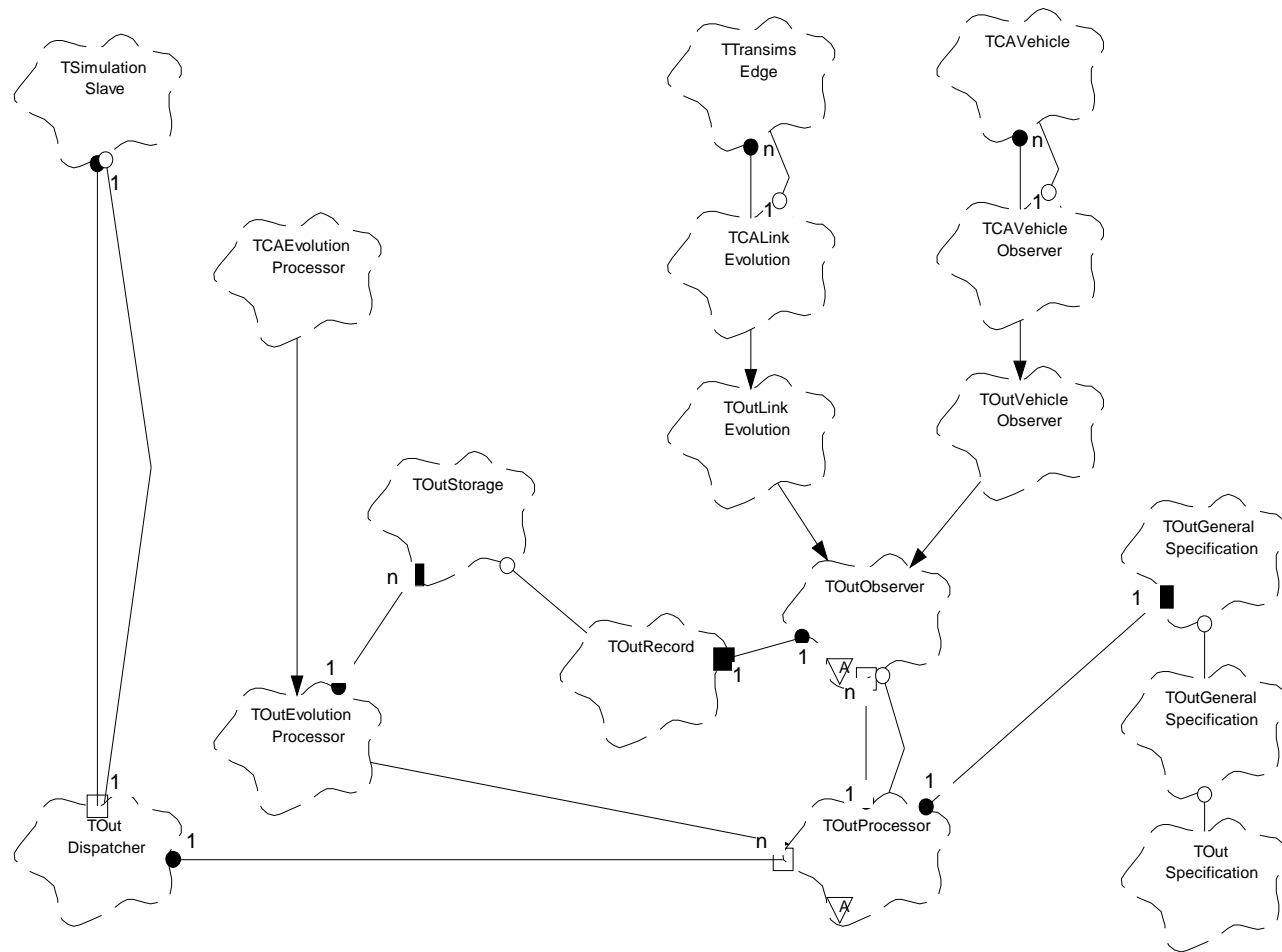


Internal Structure

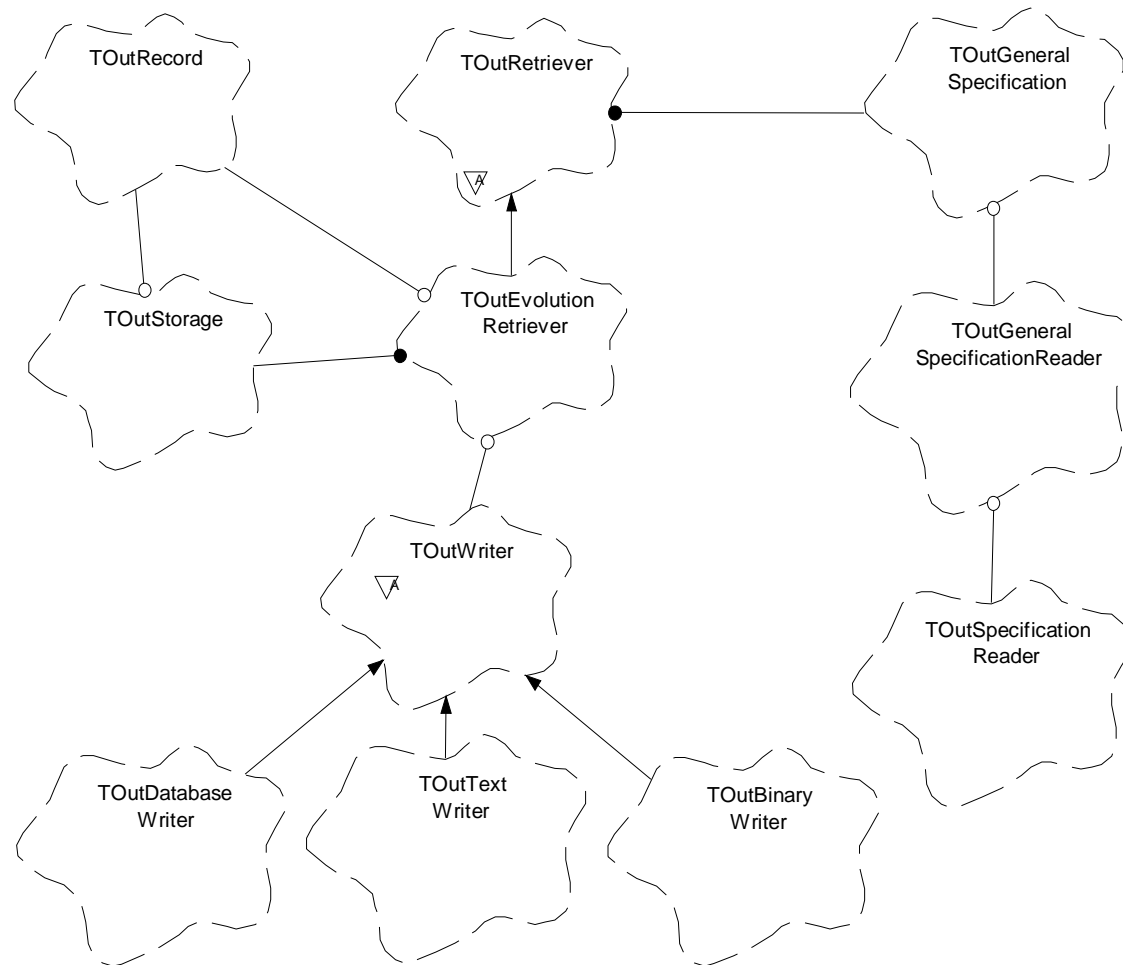
Domain Knowledge Layer



Classes Involved in Trajectory Storage



Classes Involved in Trajectory Retrieval



Modularity and Reusability

- The simulation output subsystem is not dependent on the components of TRANSIMS that use it:
 - CA microsimulation
 - Analyst toolbox-related components
- The simulation output subsystem is dependent upon only two components of TRANSIMS:
 - Network subsystem
 - Database subsystem
- The simulation output subsystem can be used to collect data from *any* TRANSIMS traffic simulator, not just the current CA-based one.

Iterative Development

The development of the simulation output subsystem has proceeded according to an iterative development process.

- √ 0. Architecture
- √ 1. Design
- √ 2. Basic functionality
- √ 3. Optimization of basic functionality
- ⇒ 4. Summary Data Processing
- 5. Enhanced disk storage

Usage

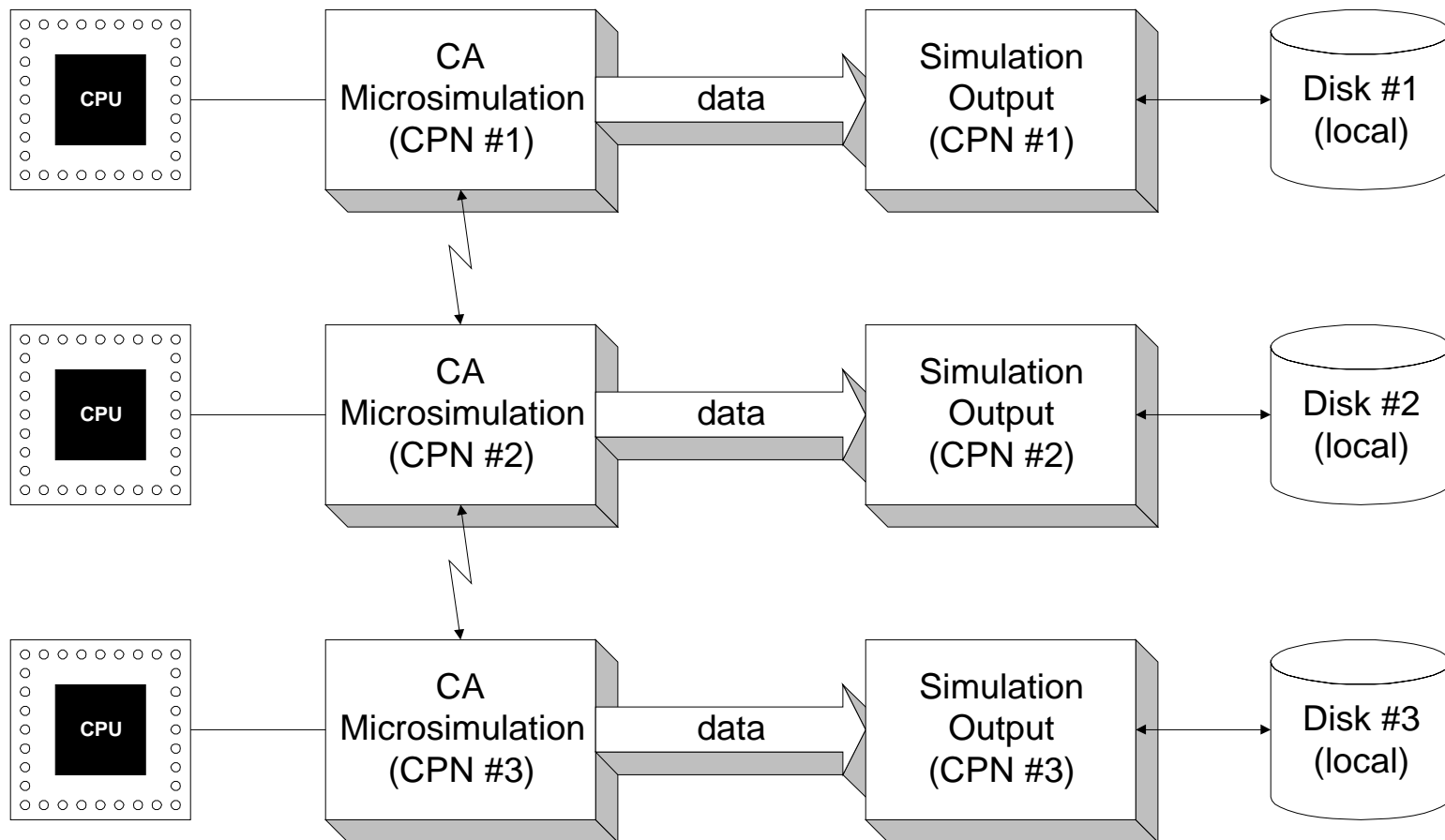
- The user specifies the following before a simulation run:
 - when and where in the traffic network to collect data
 - what data to collect
 - * raw data
 - * summaries
 - * distributions
 - * correlations
 - what data to filter out
- There are two basic types of data:
 - *Evolution data* is the trajectory information needed for animation, waterfall plots, debugging, etc.
 - *Summary data* is the statistical information needed for fundamental diagrams, measures of effectiveness, etc.

Usage (continued)

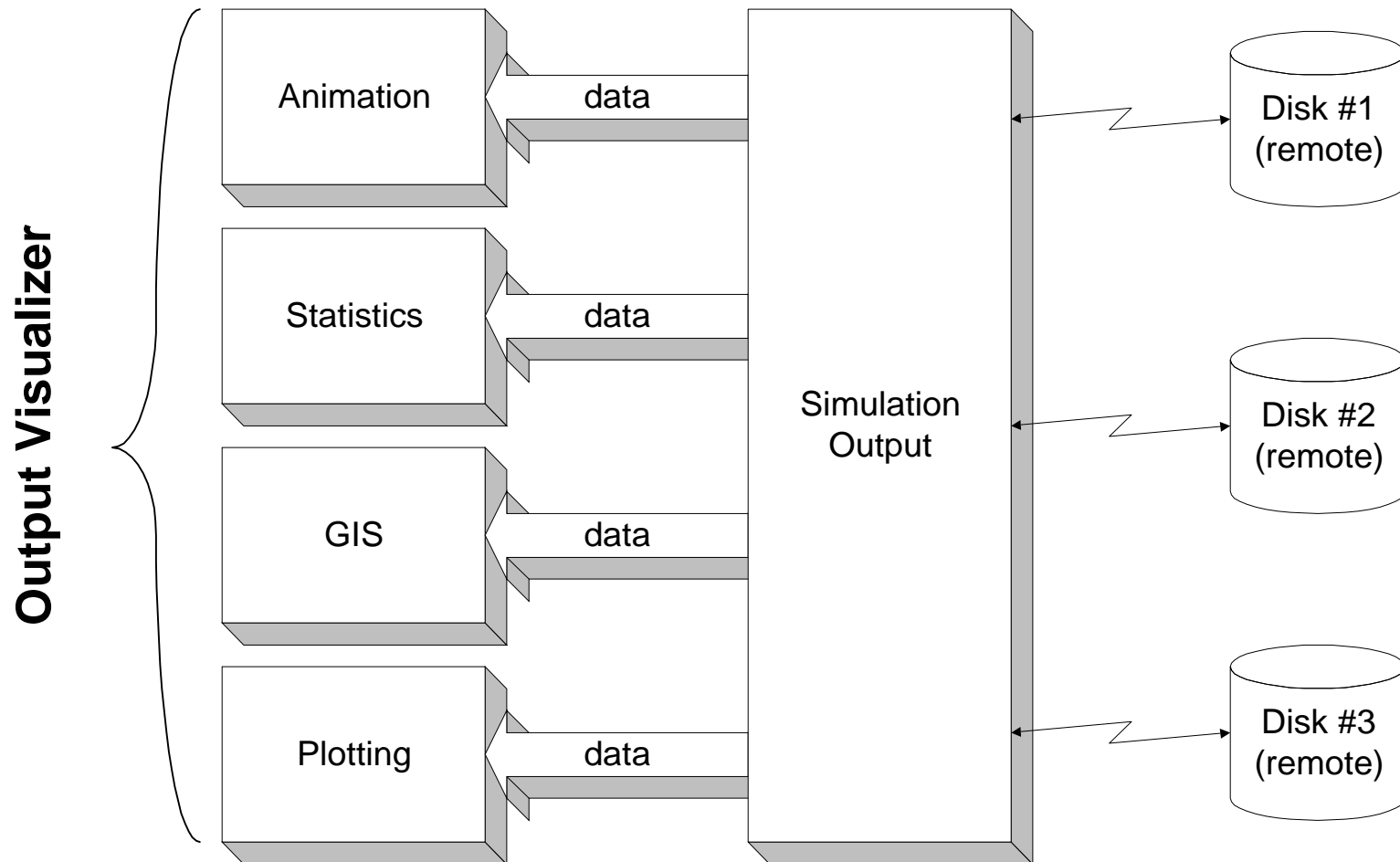
- The user specifies the following when retrieving data from a simulation run:
 - when and where in the traffic network to retrieve data
 - what data to collect
 - * raw data
 - * summaries
 - * distributions
 - * correlations
 - what data to filter out

Note that the options for collecting data from a simulation are the same as those for retrieving it from storage.

Data Collection



Data Retrieval



Current Status and Functionality

- Iterations 0, 1, 2, and 3 have been completed. This includes implementation, testing, and documentation.
- The simulation output subsystem has been linked to the CA microsimulation and collects basic “evolution” (i.e., trajectory) data.
- Data is collected locally, so as not to slow down the running simulation, but it is seamlessly accessed from the distributed file system when retrieval occurs.
- Both on collection and retrieval, the data can be filtered by time, frequency in time, node id, and link id.
- Retrieved data is stored in delimited text files (a standard import format for many commercial data analysis products).

Work Planned

- provide an ArcView interface for setting up output data collection
 - support map and spreadsheet input of collection specifications
 - add option to write CA and DumpEvolution input files
- have CA write output metadata
- enhance DumpEvolution
 - speed it up
 - improve the user interface
 - support dumping multiple specifications
 - provide diagnostic messages for missing data files
 - make it work while the CA is running
 - support the collection and retrieval of evolution data on a subset of a link rather than on the whole link
- support the collection of summary data on links^{*}
 - vehicle density, flow, velocity, count, lane occupancy
 - sample the data sparsely in space and time

^{*} We need input from users concerning the capabilities required here.

- develop a DumpSummary tool similar to DumpEvolution

Work Deferred

- indexing the simulation output data files
- replacing the NFS-based storage currently used with a more sophisticated and efficient type of distributed storage
- collecting summary data at intersections

Summary Output Issues

- What types of summaries are required for the IOC-1 case study?
 - vehicle density, flow, velocity, count, lane changes
 - lane occupancy
 - correlation matrices [?]
 - histograms [?]
- How is the data sampled for summarization?
 - duration and frequency in time
 - location on the link
 - event-driven sampling [?]
- How much flexibility is required for the IOC-1 case study data collection?
 - predefined collection strategies
 - ad-hoc collection [?]